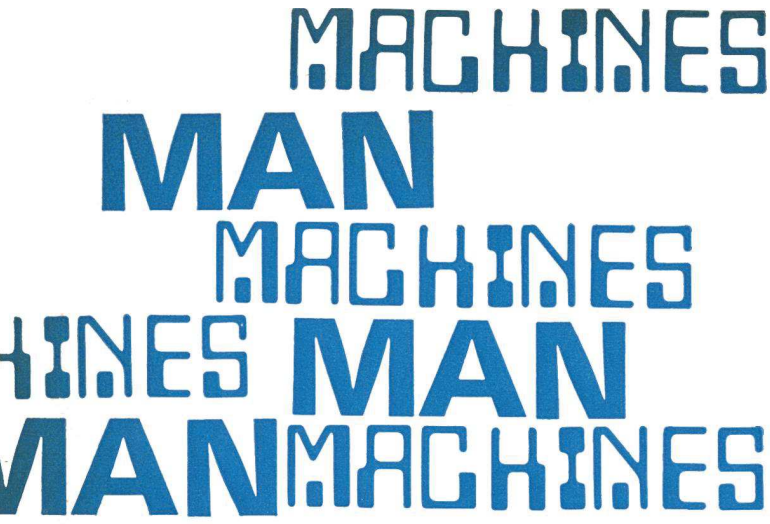

Men, machines, and

In today's computer we have a device in which meaning has its narrowest behaviorist sense—the precise linking of response with stimulus. Steps have been taken toward giving computers a picture of the world, but the results so far have been more startling than useful

> "When we listen to a person speaking or read a page of print, much of what we think we see or hear is supplied from our memory. We overlook misprints, imagining the right letters, though we see the wrong ones; and how little we actually hear, when we listen to speech, we realize when we go to a foreign theatre; for there what troubles us is not so much that we cannot understand what the actors say as that we cannot hear their words. The fact is that we hear quite as little under similar conditions at home, only our mind, being fuller of English verbal associations, supplies the requisite material for comprehension upon a much slighter auditory hint."
> —William James.[1]

Men communicate with each other by natural language and with machines by computer language. Building a bridge between these forms of communication has proved difficult, possibly because of misunderstandings about the nature and use of each kind of language. This article points out that until people succeed in storing in a computer considerable areas of human experience, they are not likely to make a computer interact usefully with natural language.

Language is a powerful tool by means of which men communicate. Computer languages are powerful tools by means of which men use machines. It has been suggested frequently that the gap between the natural languages of men and computer function should be bridged for such purposes as machine translation or information retrieval. It has proved difficult to carry such suggestions through to any useful end. This may be the result of misunderstandings concerning the nature and use of machine languages and natural languages.

The functioning of a computer is an example of meaning in a sense that would delight a behaviorist psychologist. At any moment, the physical state of the computer is described by various electric and magnetic fields, which are either positive or negative, or on or off, as well as by whether certain switches are open or closed. The computer proceeds deterministically from one state to another in a manner dictated by signals stored in its memory units, whether these be cores, tapes, or cards.

The output of the computer is likewise completely determined by the state of the computer and by its interconnections, whether they be physical or specified by the program.

The input of the computer is provided ultimately by a human being. The human being wants to make the computer process data in a way that is specified by him, to some end that he hopes to achieve. The computer certainly has a meaningful relation between the output and the input in that the output depends on the input and the internal state of the machine. The human user has a somewhat different criterion of meaning in mind. To the human user, the computer's output is meaningful in its relation both to the input—and to his intentions. He doesn't want to know about the computer's detailed and essential internal operations unless he has to. In order to get the output he desires, the human user feeds into the computer both data and a program of instructions expressed in what is called a programming language.

Even to the experienced user equipped with programming manuals that purport to tell how to make a computer system do a specified thing, the problem of using a large computing system has an element of psychological study in it. This is not the case because the meaning of instructions in a computing language is obscure to the computer, but because it may be obscure to the user. Such obscurity arises in two ways.

First, in order to make computer languages easy to learn and use, their designers make them to some degree resemble mathematics, or English, or both. Fortran, for instance, resembles mathematics and simple English. This resemblance makes Fortran easier to learn than it would be if it resembled nothing that the user had ever encountered before, either in the symbols used or in their relation to one another. However, the resemblance of Fortran to mathematics and English tempts the user to guess at the instructions needed to produce the intended output rather than to look them up in the manual. Or, even when the user does look an instruction up in the manual, he may misunderstand it. That is one way in which a user may communicate to the computer a statement whose meaning is perfectly explicit to the computer, but is misunderstood by the user.

Another source of trouble—or, if we wish, misunderstanding—arises through the complexity of controlling the computer or through malfunction, which is not infrequent. The linkage between computer input and output is a deterministic mechanism whose functioning

languages MAN MACHINES MAN MACHINES MA

John R. Pierce

Bell Telephone Laboratories, Inc.

can be described in terms of its states and the inputs. The program acts on the mechanism of the computer through several levels of language. There is the machine language, whose instructions do simple, understandable things in transferring digits in and out of specific memory locations, and in performing certain logical operations, as well as certain input and output operations, such as reading from or writing onto tapes or disks, or controlling printers.

Once removed from the machine language is the operating system and an assembly language. Together, these enable the user to cause the computer to do humanly useful but complicated sequences of operations by means of a few instructions. The operations include input and output operations and the assignment of places in the computer memory in a symbolic manner rather than by actual location. They also include certain diagnostic functions—telling a programmer that he has given a meaningless, and therefore illegal, instruction, or telling him of certain malfunctions of the machine comprising the hardware and software that make up the operating system.

Although some programs are written in assembly language, most are written in a higher-order language. By means of a compiling program the computer goes from the instructions in the higher-order language to a machine language code, which, the programmer hopes, will make the machine do what he desires.

It is not uncommon for there to be a number of layers of language, one on top of another. For instance, Trac statements may be interpreted or "translated" into Fortran statements by a program written in Fortran. The resulting Fortran program will then be compiled into assembly language, and the assembly language and the operating system will then produce the required machine language instructions to accept input, to perform the necessary logical and arithmetic manipulations, and to supply output.

Computing machines are often fallible, and hardware faults are common, but human beings are even more fallible when faced with the complicated task of programming a computer. This exists at several levels.

The fallibility of humans

One common fallibility is called errors in logical design. This means that the designer himself did not understand the hardware that he built. Sometimes what it actually does isn't what he thinks it will do. Either he didn't ask what would happen in all contingencies, or he made a mistake in judging what the hardware he designed would do. Logical faults—that is, incomplete knowledge of the basic hardware—can be overlooked and can remain obscure for a long time, because sometimes the so-called malfunction, which is really a malfunction of the designer, is apparent only under unusual circumstances.

Up through the layers of language there are other traps for the programmer. The programmer thinks he knows what he wants various instructions and sequences of instructions to do, and he thinks he knows how to make the machine carry out this intent. But often he is mistaken. The instructions always have a clear meaning to the computer; they cause it to do something completely mechanical and logical. But this may not conform to the desires or logic of the programmer. For instance, in a new computer system too close to home for the event to be entirely amusing, an innocently intended instruction in a program made the operating system inaccessible to the computer, so that the machine was inoperative for several hours. This must have been what the machine had been asked to do, but the programmer hadn't meant it that way. He didn't fully understand the nature of the machine and its operating system and its language. In a case such as this it is pretty hard to find out just where in all the complexity the mistake lies.

How does the advanced programmer deal with the problem of designing layer on layer of computer languages in such a way that he can predict the outcome of computer operation? Obviously, he doesn't, in a high-order language, try to trace the consequences of an instruction right through all the hardware of the machine. Rather, he works with what one might call levels of meaning. In writing a language over an assembly language, for instance, he assumes that he understands the implications of the assembly language completely, that is, what it will do to the machine under all conditions. Similarly, if he writes a language to be translated into Fortran, he assumes that he understands the consequences of any Fortran program in the same way that the machine will carry them out. The programmer works pretty much with native ingenuity coupled with bright ideas, many, if not all, of which can be described by the word "algorithm." An algorithm is merely a completely explicit procedure at any level for making a computer do your bidding.

Beyond ingenuity and an assortment of algorithms, the computer expert has various sorts of formal mathematics, which can help him to see the consequences of his actions. Presumably, the Lord God Almighty can see immediately the consequences of any assumption or action, and so he doesn't need the notation and the opera-

tions of mathematics in order to help him trace consequences out. Man is pretty poor at seeing consequences. Thus, to the hardware man, Boolean algebra can be of great help, for it enables him, through mechanical mathematical operations, to realize certain logical functions of input economically by means of specific hardware.

On the more complicated levels of overall machine design and of programming, there are also some useful mathematical tools. These include automata theory and mathematical linguistics. By giving simplified models of the essential aspects of a computer, automata theory tells one what one can do with machines, and hopefully gives a clue as to how this can be accomplished. Mathematical linguistics teaches how to formulate rules of syntax in the most elegant manner. It is an aid in designing languages so that in going from one level of language to another the computer will more often do something that was intended by the programmer rather than something that he accidentally or involuntarily put into the program without understanding its nature and consequences.

Although it is easy to get disconnected from this reality, it is clear that reality in the computer is meaning in the sense of explicit physical events and sequences of events in the mechanism of the machine and the output that these produce. In the solving of even a simple problem these events are simple in themselves and in their immediate interactions, but the chain of events is too long and highly interrelated for a human being to grasp. The problem of the programmer and the program designer is to present the user with a language that is just as meaningful, in that a given input always has a precise output, but a language of which the programmer can grasp the consequences. Thus, a desired input language is one in which the programmer seldom makes the mistake of asking for something he really didn't mean, although he doesn't know just what the computer does step by step in carrying out his intent.

There can scarcely be one ideal programming language; programming languages are designed for various problem areas. And within a sizable problem area, a single, ideal programming language that is perfectly understandable with a modicum of study is perhaps unattainable. Even if we could attain it, programmers would still be so fallible as to make mistakes. Thus, in a chosen field, one approaches some practical level of usefulness in which programs usually do what one would expect after a reasonable study of the programming manual, and in which the language is so well adapted to human needs in a usefully broad area that the programmer seldom makes mistakes in terms of what is written in the manual. It is certainly desirable to study ways of writing languages whose consequences can be completely described in a manual of reasonable length, but this is obviously not necessary for the satisfactory use of computers, and it has scarcely been attained at the present time.

I have now come almost to the end of what I want to say concerning computers, computer languages, and the use of these languages and computers by human beings, and here I would like to philosophize a little.

First, a behavioristic sort of meaning is paramount in dealing with computers. The computer always does something. What it does depends on how it was built—whether the builder understood it or not. The programmer has some intent that seems clear to him. If it is clear to him and the computer doesn't understand him (that is, it doesn't process his data as he wished it to), the fault is either that he didn't understand the manual, or that the manual was at fault and failed to predict what the consequences of a given program would be, either through a failure of software design at some level or through a logical error of the designer of the machine.

In computer language, grammar and syntax are tools in relating the operation of various parts of the computer to sequences of humanly useful symbols. It may be that human beings are worthy of study for as long as man exists, because as long as man exists, human beings will surround him. Computers are worthy of study because they do useful things for man and because man wants to make better computers that will do even more useful things. Hence, in human terms, the mathematical tools necessary in the design and use of computers are important only insofar as they enable us to design better computers or to use computers better. Few mathematicians are interested in improving real computers. Mathematicians do, however, become fascinated by the branches of mathematics that seem to be pertinent to computers, and through this fascination they may sometimes produce results useful to computer designers and programmers—but they may not. Those interested in designing, understanding, and using computers will cast a wary eye at the mathematically inclined, and ask whether a given piece of work has really advanced our ability to build and control computers.

MAN MACHINES

So far I haven't said anything about artificial intelligence, and I don't intend to say much. Many early computer enthusiasts thought that computers should resemble human beings and be good at exactly the tasks that human beings are good at. This is like drawing a vehicle by a steam man between the shafts, or designing an airplane that will light on a tree. It is facing the future with one's back squarely toward it. Progress with computers has come through employing their useful potentialities, not through trying to make them conform to human behavior. The new and useful tasks that computers do are often far more complicated than the useful tasks computers did in the past. Hence, computers continually look more "intelligent." It would be foolish to set any limit to the complexity of the tasks for which computers may some day prove useful. But the sort of performance the artificial intelligence enthusiasts have worked toward hasn't proved useful. Computers have been miserable at theorem proving, music composing (at least we don't enjoy their compositions—maybe computers do), chess playing, and general pattern recognition. In view of the tremendous usefulness of computers, all I can say about the failures in these fields is, so what?

Another aspect of artificial intelligence I might describe as the easy way out. Some people believe that by combining an element of randomness with a so-called heuristic strategy, one can get the computer to solve problems better than through an intensive exercise of human

ingenuity toward a particular goal. It just hasn't worked out that way.

The human use of human language

We now come to a quite different field, the human use of natural or human language. Here it is hard to get started. Although we all have implicit knowledge of human language, in that we are able to produce, use, and understand it, no one understands human language explicitly, as we can, at least in principle, understand a computer language in connection with a computer. The human world of our apprehension, thought, and action is in computer terms a high-level world. It is clearly many levels removed from neuroanatomy and neurophysiology. Our understanding of neurophysiology and neuroanatomy is small but increasing, but we are certainly not at such a point that we can either work upward and explain higher-level function explicitly in terms of function at the neural level or work downward and find out just how high-level instructions result in appropriate physical actions. Most human behavior, including language function, has to be tackled at a higher level.

It is tempting to liken the psychologist to a computer programmer equipped with a very imperfect programming manual, and with no really useful insight into the internal structure of the machine. Being denied access to the machinery, the psychologist may seek understanding in terms of input and output. His manual relating these includes experimental results, common knowledge, and folklore. It is perforce incomplete, and it is almost certainly wrong. This wrongness may in some instances be of a very fundamental kind, or it may be a wrongness engendered simply by the fact that different human beings are different whereas different computers of the same model are presumably the same.

If this were all there was to it, a study of human linguistic behavior (or, indeed, of human behavior) might well start with a search for the syntax of human language, to be followed by an exploration of the effects of well-structured sentences on human beings. Alas, things aren't that easy.

My attention was first drawn to linguistic problems through Claude Shannon, with whom I was closely associated while he was inventing information theory. Information theory tells us that messages from a stochastic source can be efficiently encoded if we take into account the probabilistic constraints on the source in producing symbols. This led Shannon to produce, by probabilistic means, sequences of symbols that resemble English words, and sequences of words that resemble English text, as shown in the following examples:

1. Zero-order approximation (symbols independent and equiprobable):

XFOML RXKHRJFFJUJ ZLPWCFWKCYJ FFJEYVKCQSGHYD QPAAMKBZAACIBZLHJQD.

2. First-order approximation (symbols independent but with frequencies of English text):

OCRO HLI RGWR NMIELWIS EU LL NBNESEBYA TH EEI ALHENHTTPA OOBTTVA NAH BRL.

3. Second-order approximation (digram structure as in English):

ON IE ANTSOUTINYS ARE T INCTORE ST BE S DEAMY ACHIN D ILONASIVE TUCOOWE AT TEASONARE FUSO TIZIN ANDY TOBE SEACE CTISBE.

4. Third-order approximation (trigram structure as in English):

IN NO IST LAT WHEY CRATICT FROURE BIRS GROCID PONDENOME OF DEMONSTURES OF THE REPTAGIN IS REGOACTIONA OF CRE.

5. First-order word approximation. Rather than continue with tetragram, . . . , n-gram structure, it is easier and better to jump at this point to word units. Here words are chosen independently but with their appropriate frequencies.

REPRESENTING AND SPEEDILY IS AN GOOD APT OR COME CAN DIFFERENT NATURAL HERE HE THE A IN CAME THE TO OF TO EXPERT GRAY COME TO FURNISHES THE LINE MESSAGE HAD BE THESE.

6. Second-order word approximation. The word transition probabilities are correct but no further structure is included.

THE HEAD AND IN FRONTAL ATTACK ON AN ENGLISH WRITER THAT THE CHARACTER OF THIS POINT IS THEREFORE ANOTHER METHOD FOR THE LETTERS THAT THE TIME OF WHO EVER TOLD THE PROBLEM FOR AN UNEXPECTED.

From the point of view of information theory, it is interesting that there are probabilistic constraints on English text. By observing these constraints, we can encode English text for economical transmission. But it was easy to jump from Shannon's experiments to false or unrewarding conclusions. Because Shannon considered probabilistic models that were Markov processes or else finite-state machines, one might conclude that a Markov process or a finite-state machine could, in principle, produce any English sentence. Chomsky denounced the fallacy of any such models almost before they were proposed.[2]

There was another unrewarding conclusion that Shannon's experiments made it easy to jump to. This was that somehow syntactic rules must be not only necessary to, but central to, the human use and understanding of language. Linguists set themselves the task of devising a grammar that would produce all well-formed (I guess that means "correct") English sentences and would not produce any but well-formed English sentences. Whatever may be said concerning the ultimate success or failure of such a program, the real question is that of its relevance to the human use of language.

Here we must distinguish several areas of importance. One, for instance, is the teaching of a language to adult foreigners. Successful teachers will say that it is important to give a student the sounds of a language and its structure, and that he will then acquire its vocabulary.

But the structure that a teacher gives to his students is not a complete description of how to produce all well-formed utterances and none but well-formed utterances. This appears to be beyond the power of the most complicated grammars that have been devised, for English, for instance. And those grammars are so complicated that they are of little use in teaching English.

The grammar the language teacher needs is rather like tips for playing golf or tennis, or rules for spelling.

What is needed is reminders that can be remembered and used by human beings; that is, useful rules to supplement and guide the acquisition of an implicit knowledge of the language.

It isn't easy to arrive at such an incomplete grammar. We all know foreigners who, though capable of quoting endless grammatical rules, make mistakes in speaking English. However, many natives who can enunciate few if any grammatical rules can speak flawless English.

Another possible use of a grammar of the English language would be in instructing computers how to do things. It is said that modern linguistics has contributed greatly to the writing of compilers and the devising of machine languages. If this is so, it shows merely that syntax is very important in machine languages. Advanced linguistics has not enabled computers to process natural language in the sense of translating from one language into another, or of acting on instructions given in natural language. The reason for this is easy to see.

It is plausible that meaning rather than grammar is the essential element in the human use of human language. No one can argue that grammar isn't there. However, I can say that I've heard a dean of engineering talk perfectly sensibly and understandably for half an hour without uttering a single well-formed sentence. I can also argue that efforts such as that of Katz and Fodor[3] to bring meaning in through the back door of grammar seem to lead more to increased complexity than to added enlightenment.

The machine as translator

The failure of a grammatical approach to the problems of natural language is well illustrated in attempts at machine translation. Here one can go a considerable distance in translating from one language to another closely related language simply through dictionary lookup. There can be a good deal of intelligibility in the output, but it is poor in two ways. Because many words are used in several senses or have several meanings, through dictionary lookup one often gets either the wrong word or an assortment of words. Coupled with this is a strange word order in the output which is often baffling.

It may be possible to learn to make some sense of text translated simply through dictionary lookup, but one would like to do better. The approach of machine-translation people has been to try to select words and to amend the word order by using grammatical constraints. In any practical sense this hasn't worked. Machine translation still is chiefly a sort of gobbledygook. The trouble becomes clear when we turn from machine translation to machine parsing of sentences.

The machine parsing of English sentences by Oettinger and his associates,[4] as well as by others, has shown that almost all English sentences are syntactically ambiguous, even with the most powerful, unwieldy, and generally useless grammars that linguists have been able to devise. Thus, we all know instantly the meaning of "Time flies like an arrow," but a computer may well conclude that there are time flies who like an arrow, or that someone is being instructed to time flies in the same manner as an arrow would time flies. There are other grammatically sensible interpretations of this sentence as well. The diehards ceaselessly try to amend their grammars with new rules and with semantic markers. Some disgusted spectators have turned in other directions.

Experiments by Miller and his associates have shown that sentences formed according to syntactic rules which result in a structure resembling that of natural English sentences are more easily understood than sentences formed according to equally rigid rules that give sentences that don't much resemble English sentence structure. This is not surprising. Experiments by McMahon, Slobin, and Gough[5] also seemed to show sentences with a simple grammatical structure to be more quickly apprehended than sentences that conveyed what seemed to be the same meaning through a more complicated grammatical structure, such as the passive voice. This opened up an interesting line of experimentation, but further experiments should make one wary of conclusions based on grammatical theories alone.

Indeed, human judgment in the area of grammar seems to be influenced more by meaning than by syntactical structure. Some years ago S. M. Pfafflin[6] carried out an experiment at the Bell Telephone Laboratories with computer-generated sentences of simple and acceptable syntactic structure. She asked naive subjects whether or not these sentences were "grammatical." The subjects clearly based their judgments on meaning as well as on syntax. If they could make sense of a sentence they judged it to be grammatical; if not, not. The more intelligent and ingenious subjects could think of and accept more unusual meanings than duller subjects, and they judged more sentences to be grammatical. This makes it seem plausible that the idea "grammatically correct" has meaning and use to ordinary human beings only in connection with meaningful sentences.

This corresponds to my introspective view of my reaction to a sentence, or even a nonsentence, of human origin. My immediate reaction is, what does he mean? When human, or indeed computer speech, is poorly intelligible, my mind races to attach a meaning to the utterance, and when I succeed the sounds become "clear." When I try to parse a sentence, I already know what the sentence is saying—or I can't parse it.

Thus, it appears that syntax does not play the same sort of role in the human use of human language that it plays in the computer use of computer language. It is natural for a computer to parse a statement as a very first step in making use of it. A man cannot parse a statement in natural language until he knows something about the meaning of the statement. Further, errors in syntax usually change the meaning (to the computer) of a statement, but they do not necessarily change the meaning to a human being of a statement in natural language.

No doubt syntax comes into the understanding of speech, but meaning seems somehow to come first. And this meaning includes our knowing what subject is being talked about and knowing something about the subject, not just context in the sense of association of words in a sentence or on a page.

In experiments carried out at the Bell Telephone Laboratories by J. S. Sachs,[7] subjects who were queried about material they had heard or read remembered chiefly the content rather than the grammatical structure or the choice of words. This was true even when they were presented with a statement embodying the content of the last sentence they had been exposed to and were asked whether or not this was identical with that last sentence.

Subjects were prone to say that the test statement was identical with the last sentence if the meaning was the same, even though some words had been replaced by synonyms.

Where do we go from here?

It is all very well to say that meaning is central to the use of language, but this tells us neither how to define meaning, nor how to make use of it. Where do we go from here? That depends on what we want to do.

If we want to use natural language as an input to machines, it means that in some very real sense the machine must understand what the language is saying in the same sense that a human being does. This is the position that V. K. Yngve reached after years of work in computational linguistics. As he has written,[8] "Work in mechanical translation has come up against a semantic barrier. We have come face to face with the realization that we will only have adequate mechanical translation when the machine can 'understand' what it is translating and this will be a very difficult task indeed . . ."

Human beings get at the content of sentences because they have an internal map of the world to which the sentences are relevant. This somehow enables them to arrive at the content of the sentence whatever its grammatical form, and even when it is ungrammatical. As I have already observed, it even enables a person to arrive at the content of a sentence when some words are not clearly enunciated or clearly heard. This is because the person somehow checks the sentence against what he already knows about the world or the subject under discussion.

Fred Thompson[9] was able to incorporate a little of this in DEACON, a machine system for giving naval information in response to simple English inquiries. For instance, DEACON is able to understand that in the phrase "Boston Navy Yard," the meaning is not the yard of the Boston Navy, but the Navy Yard at Boston. The machine arrives at this conclusion, not through linguistics, but because someone has stored in it the information that there is a Navy Yard at Boston whereas Boston Navy is an undefined and therefore meaningless term.

We may hope that in the future it will be possible in some way to give machines enough knowledge of a subject other than their own construction to enable them correctly to interpret natural-language statements in some limited field of interest.

Indeed, this has been done in some degree in many modern computing languages and operating systems. The computer will accept statements with some range of syntax and interpret them plausibly. The computer has stored away in it humanly useful responses to certain errors in programming to which human beings are generally prone. These include such admonitions as "line 10 is not a statement" (which means it is not syntactically allowable); "in line 15, *N* is not defined" (which means nobody has put in a value for it or an equation defining it); "what" (which means that the user has done something wrong at that place in the program).

The problem of enabling a computer to use natural language as an input by incorporating in it some sort of map of even a portion of the world is difficult and ill-understood. Man's knowledge of the world enables him to make use of ungrammatical sentences when the meaning is clear. Man can also interpret correctly, that is, give correct operational meaning to, sentences that use words in a "broad" sense. Thus, we have no trouble with "open the door," "open the box," "open the fan," even though the particular physical operations are different. Moreover, if I were handed some strange, unnameable but "openable" object and told to open it, I believe I would have a fair chance of succeeding.

It almost seems as if man uses an ability to coin apt but (to him) new phrases or word usages, which, in a context of particular things and actions, will almost always be terpreted correctly. Thus, "open" seems to be associated with making the inside accessible from the outside, and such words as "cut" and "tear" with actions that can be means of providing such access. "Open the jalousie," or "tear" or "cut the ottoman open" are interpreted easily enough when such objects are before us or known to us, whether or not we have previously associated the word "open" or the idea of opening with them.

However he is able to do it, somehow, man can respond to instructions phrased in broad concepts. If the computer is to respond to the same sorts of instructions that man responds to, it must somehow perform this miracle.

Today we have in the computer a device in which meaning has its narrowest behaviorist sense: the precise linking of output with input, of response with stimulus. The computer is dependent on and susceptible to syntax. The computer has little or no picture of a world outside itself, to use in testing or interpreting statements or in ascertaining the precise meanings of ambiguous words.

Man, who lives in the world and has some knowledge of it, jumps at content or meaning before he fully resolves syntax, if indeed he does, and he interprets words such as "open" correctly in a wide variety of contexts.

Some steps have been taken toward giving computers a picture of some portion of the world.[10] So far, the results have been more startling than useful. Someone may well go much further and be able to put into computers maps of considerable areas of human experience or human knowledge. But until people succeed in doing this in a satisfactory way, they are not likely to succeed very well in making computers interact usefully with natural languages.

REFERENCES

1. James, William, *Talks to Teachers on Psychology and to Students on Some of Life's Ideals.* New York: Holt, 1899, p. 159.
2. Chomsky, N., "Three models for the description of language," *IRE Trans. Information Theory*, vol. IT-2, pp. 113–124, 1956.
3. Katz, J. J., and Fodor, J. A., "The structure of semantic theory," *Language*, vol. 39, pp. 170–210, 1963.
4. Kuno, S., and Oettinger, A. G., "Syntactic structure and ambiguity of English," *Proc. AFIPS*, Fall Joint Computer Conf., vol. 24, pp. 397–418, 1963.
5. McMahon, L. E., "Syntactic analysis as a part of understanding a sentence," unpublished doctoral dissertation, Harvard University, 1963.
6. Pfafflin, S. M., "Grammatical judgments of computer-generated word sequences," presented at Seventh Annual Conf. on Linguistics, New York Linguistic Circle, Dec. 1961.
7. Sachs, J. S., "Recognition memory for syntactical and semantic aspects of connected discourse," *Perception and Psycholinguistics*, vol. 2, pp. 437–442, 1967.
8. Yngve, V. K., "Implications of mechanical translation research," *Proc. Am. Phil. Soc.*, vol. 108, pp. 275–281, 1964.
9. Craig, J. A., Berezner, S. C., Corney, H. C., and Longyear, C. R., "DEACON: Directed English Access and CONtrol," *Proc. AFIPS*, Fall Joint Computer Conference, vol. 29, pp. 365–380, 1966.
10. Weizenbaum, J., "Contextual understanding by computers," in *Recognizing Patterns*, P. A. Kolers and Murray Eden, eds. Cambridge, Mass.: The M.I.T. Press, 1968, pp. 170–193.